The Five Tribes of Machine Learning

Antti Halme

The Apartment @ Portobello Star
12 November 2019

Agenda

- Hello, Setup (19:00)
- **Introduction** (19:30-ish)
- Housekeeping
- **Symbolists, Connectionists, Evolutionaries**

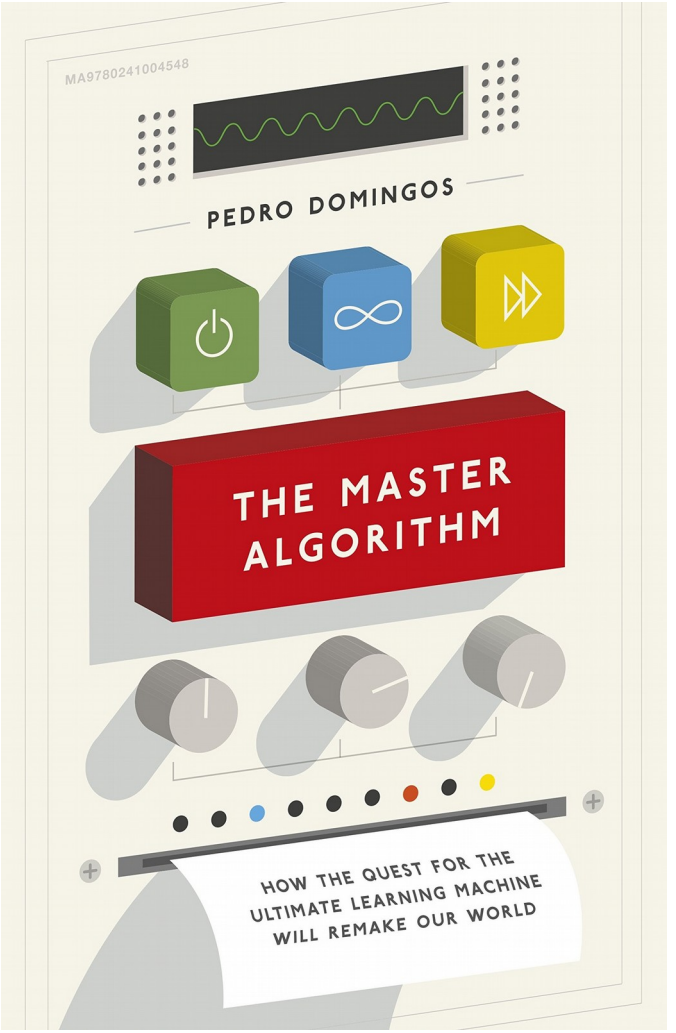
~break~

- **Bayesians, Analogizers**
- **Big Picture**
- **Discussion**
- Finish (21:30)

The Book

- ***The Master Algorithm***
  - ***How the Quest for the Ultimate Learning Machine Will Remake Our World***
  - 2015
  - ISBN 9780141979243 (paperback)

- **Prof. Pedro Domingos**
  - University of Washington
  - ACM SIGKDD Innovation Award
  - IJCAI John McCarthy Award
  - AAAI Fellow, NSF Career Award, Sloan Fellow, Fulbright Scholar, IBM Faculty Award
  - **~Bayesian; recognised for contributions to unification of logic and probability**



Introduction

- **Machine Learning**
  - Have the computer write the program!
    - Faster, more efficient, more accurate
  - Science of systems that learn from data
  - Different learners make different assumptions, are useful for certain things, but not others
  - “Tribes”, the major schools of thought within ML

Introduction

- **Book Thesis**
  - General purpose learning system could be within reach
  - *The Master Algorithm* is a call to arms
    - A whirlwind tour of techniques, a fairly accessible account of the history of machine learning
  - **Master Algorithm hypothesis:**

    “All knowledge — past, present, and future — can be derived from data by a single, universal learning algorithm.”

Introduction

> *"Each tribe has a set of core beliefs, and a particular problem it cares about the most. It has found a solution [..] and it has a master algorithm that embodies it."*

- **The Five Tribes**
  - The Symbolists — logic, philosophy
  - The Connectionists — neuroscience
  - The Evolutionaries — evolutionary biology
  - The Bayesians — statistics, probability
  - The Analogizers — psychology (?) (language?)
- *Of course, the reality is a bit more subtle*

Housekeeping

The Symbolists

- **All intelligence is symbol manipulation**
- Questions as equations, answers via symbol shunting
- Learning is built on existing knowledge
  - New knowledge created by operating on existing data

- **The old school tribe**
  - Shares history with early AI
  - A knowledge engineering legacy
  - Old mathematics, well understood

The Symbolists: Inverse deduction

- Identifying and extracting regularities captured by data
  - Propositional logic
  - **Rules** form: A ⇒ B
- **Inverse deduction**
  - Identifying missing components that block deductive reasoning
  - New knowledge created through generalisation
  - *Reasonable* operators fill the blanks, inherently uncertain
  - *Non-standard definition?*

Deduction:

Socrates is human.

All humans are mortal.

Therefore, _____

Induction:

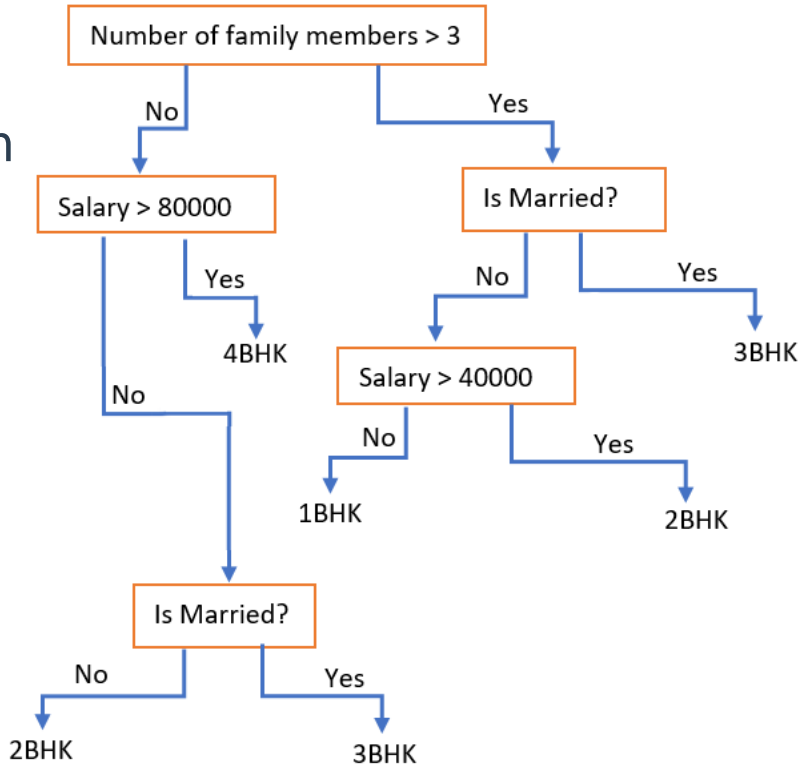
Socrates is human.

_____

Therefore, Socrates is mortal.

The Symbolists: Decision Trees

- **Decision trees**
  - Knowledge as a series of choices, path from root to leaf is a rule: highly interpretable
  - Multiple rule sets can match an instance
  - Classifier that solves ambiguity problem through “a game of twenty questions”
- **Learning through Divide&Conquer**
  - Pick low-entropy test attribute
  - Divide set until all samples agree on label
- **Random forests**
  - A powerful ensemble extension
  - Create many trees, select representative or ”average” tree for use in classification



fast.ai

Making neural nets uncool again

The Symbolists: Knowledge engineering

- **Symbolists share history with knowledge engineering**
  - Dominant AI school in the 1970s, 1980s
  - Failed to deliver on AI hype

    → *Learning from data proved much easier*
- **Knowledge acquisition bottleneck**
  - Extracting knowledge from experts and encoding it as rules proved too difficult, too laborious, too unreliable
- **See**
  - CyC — *the* knowledge base
  - Prolog, Coq — logic programming, proof assistants

The Symbolists: Conclusion

- **Oldest tribe**
  - Knowledge engineering and AI history
  - Ancient theory, well understood formal machinery
  - Supercharging with data and modern compute?
- **Inverse deduction**
  - Identify missing components, generalise
  - Highly general purpose, easily confused by noise
  - Induction space vast and hard to navigate
  - Hopelessly discrete: Many things are not black and white
- **Decision trees**
  - Knowledge as a series of choices
  - Highly interpretable, inefficient knowledge encoding
  - Random forests, a powerful ensemble extension

The Connectionists

- **Learning is what the brain does**
- Knowledge is stored in connections between neurons
  - Learning is about tuning these weights
- Which weights are responsible for which errors?

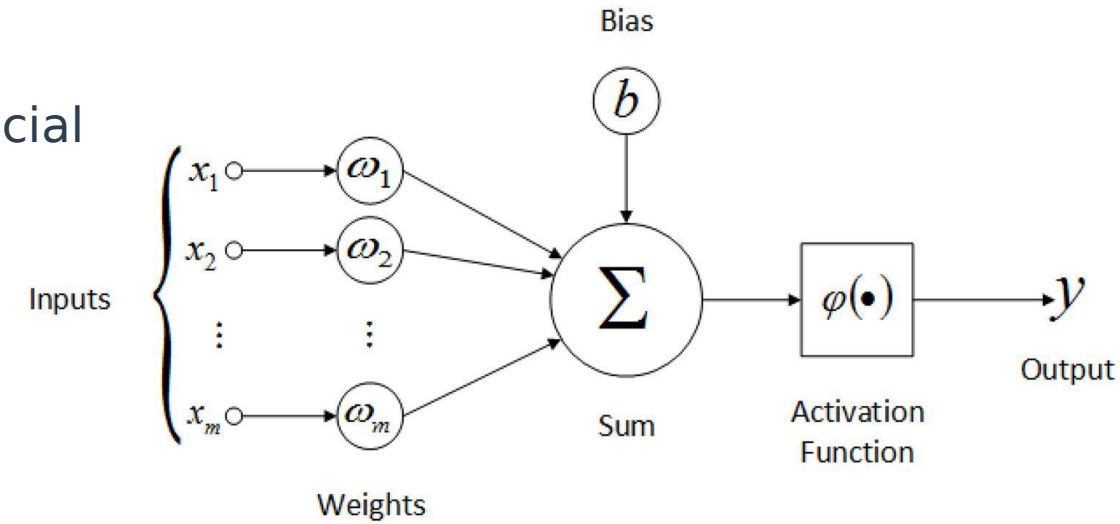
- **The dominant tribe today**

The Connectionists: Neurological Basis

- **Hebb's Rule:** *"Neurons that fire together, wire together."*
- Knowledge is not local, but diffuse in the network
- Learning is about comparing output and reality and adjusting accordingly

- *Perceptron (1958)*
  - Mathematical model of the artificial neuron, inspired by the brain
  - One layer of weights, binary threshold activation function
  - Weighted vote describes a hyperplane in input space

  → Multi-layer perceptrons



The Connectionists: Learning Weights

- **Credit Assignment Problem:** If we layer neurons, how do we know which hidden layer node weights to nudge when learning?
- Hopfield nets (1982) → Boltzmann machines (1985)
- **Backpropagation** (breakthrough in 1986)
  - An efficient way to do gradient descent optimisation in a multilayer perceptron
  - A way to pass error information to hidden layers

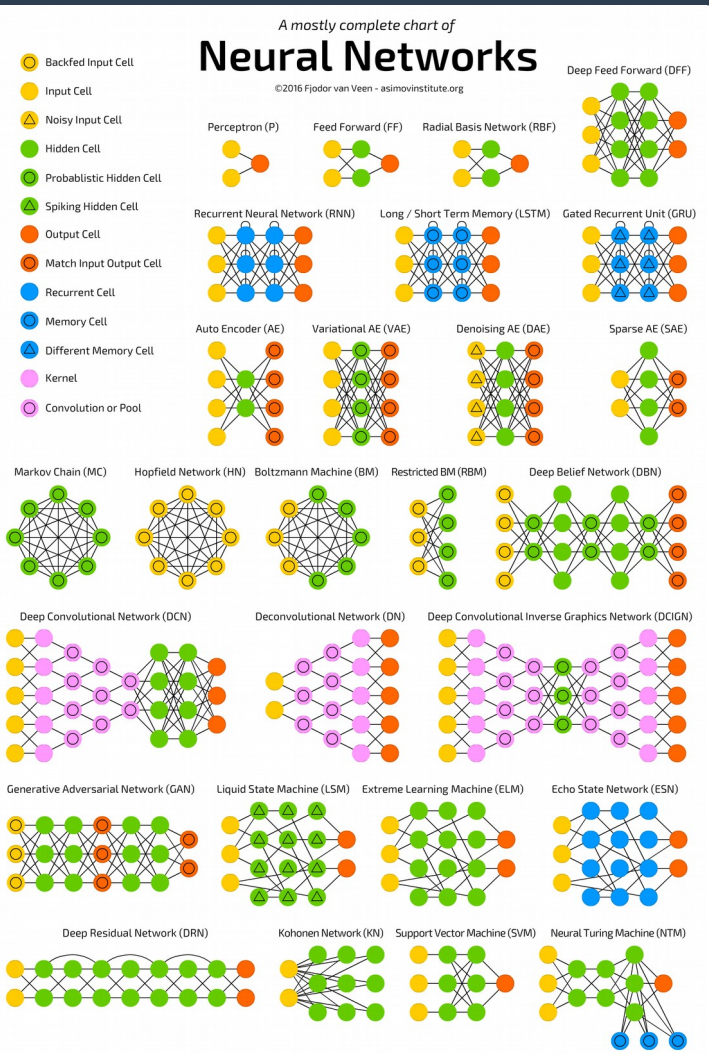
    → Solves the credit assignment problem
  - Started the neural networks renaissance

The Connectionists: Neural Network Zoo

- Backprop can easily handle a few hidden layers, larger networks are trickier to train
- Name of the game is designing custom networks with specific applications in mind, tweaking network data flows

  → **Neural Network Zoo**

- Examples:
  - Autoencoders, RNNs, LSTM, Convolutional NNs, GANs
  - Deep Learning ~ multiple hidden layers



The Connectionists: Conclusion

- Connectionists are reverse-engineering the brain by building computation inspired by the network structures of biological neurons
- Knowledge lives in the weights
- Learning, inspired by Hebb’s rule, is a matter of adjusting weights
- More layers, non-linear activations → learn more complex functions
- Backpropagation key breakthrough, solved credit assignment problem
- Currently hottest tribe, moving forward at a crazy pace
- Exciting results, wide application areas

The Evolutionaries

- **Learning is all about natural selection**
  - Simulate the evolutionary process, build anything
- Learning structure is the real challenge
  - Parameter tuning can then follow
- **The underdog tribe**

The Evolutionaries: Nature's Algorithm

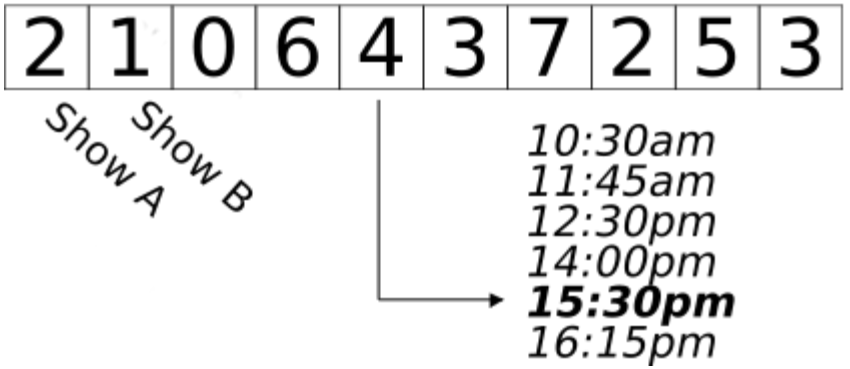
- **Genetic Algorithm**
  - John Holland, 1960s: A "population" of "genes" interacting, a disorderly **search** for "fitness"
  - **Fitness function:** Give a candidate program a numeric score to measure fitness for purpose (Human systems only!)

- System analogy of sexual reproduction
  - **Gene:** Encoded data; a set of instructions, parameters for a process
  - **Population:** Pool of genes, weighted to favour fit genes
  - **Crossover:** exchange of data in which two genes split and recombine as offspring
  - **Mutation:** point change in a data location in a gene

The Evolutionaries: Case *Fringeling*

- **A real world GA example** — Festival Programming!
  - A little hobby project of mine using GeneticJS (unmaintained)
- Objective: Build a festival programme based on preferences
  - For a set of liked shows (& times), find a schedule that maximises the number of shows seen over a multi-day visit, avoids overlap
  - Secondary: minimise travel time, minimise total distance, leave buffers before/after show, leave time for lunch, etc.

The Fringeling Gene

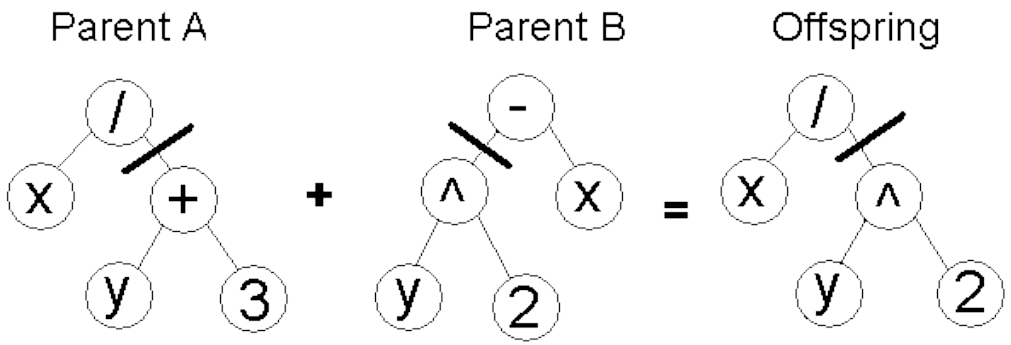


The Evolutionaries: Genetic Processing

- **Exploration-exploitation dilemma:** If you've found something that kind of works, should you keep investing more in that, or should you try new things in hope of improvement?
- **Schema theory:** Each successful gene is a building block for future genes: process is not random, the combinatorics is working *for* you
- **Benefit of crossover** an open question
  - With just mutations and a large population, GA ~ hill-climbing
  - Does sex optimise for “mixability” or maybe robustness?
- **Nature vs. Nurture:** Evolution slow, culture fast!
  - Evaluating complex gene fitness is slow
- **Baldwin effect:** Learned behaviours become genetically hardwired

The Evolutionaries: Genetic Programming

- John Koza: What if we could evolve not just parameters, but the whole process?



- **Genetic Programming**
  - *Assemble* the best sequence of subroutines and instructions
  - Programs are trees of subroutine calls, crossover at subtree level turns one program into another
  - Start with a population of random programs. Make use of crossover, mutation and survival to gradually evolve better programs until tests pass.

The Evolutionaries: Conclusion

- Simulating natural selection is a nice compromise solution to the exploration-exploitation dilemma
- Genetic algorithms and programs excel in learning structure
  - “Genes” encode candidate solutions
  - Fitness function ranks the gene pool for selection
  - Over many generations, genes mutate and combine to form ever better solutions from building blocks
- Genetic programming skips parameter encoding
  - Evolution at the functional program subtree level
  - Even more expensive to evaluate, but more expressive

The Bayesians

- **The main concern is uncertainty**
  - Learning is uncertain inference
  - Learning is model selection based on data
- The challenge is to deal with noisy, incomplete and even contradictory information
- Bayes’ theorem tells us how to update our beliefs in light of new evidence
- **The tribe with the sophisticated tools**

The Bayesians: Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- A rule for updating your beliefs based on new data
  - If evidence agrees with hypothesis, P(hypothesis) goes up
  - How to learn efficiently over all the data, across all models?
- Formula is easy, finding the probabilities is the job
  - For frequentists, statistics is the only way
  - For Bayesians, subjective estimates are the way to go

The Bayesians: Deriving the Theorem

- **Principle of indifference:** In the absence of any relevant evidence, belief in all outcomes under consideration should be distributed equally. (Epistemic probability)
- **Prior probability:** (Subjective, intuitive) probability of an event, before seeing any evidence.
- **Posterior probability:** Probability after weighing evidence
- **Conditional probability:** Probability of an event, given another event. P(A|B) : probability of event A, given event B has occurred.

The Bayesians: Deriving the Theorem

$$P(cause|effect) = \frac{P(cause) * P(effect|cause)}{P(effect)}$$

- "The more likely the effect, given the cause, the more likely the cause given the effect."

$$\blacktriangle\ P(effect|cause) \Rightarrow \blacktriangle\ P(cause|effect)$$

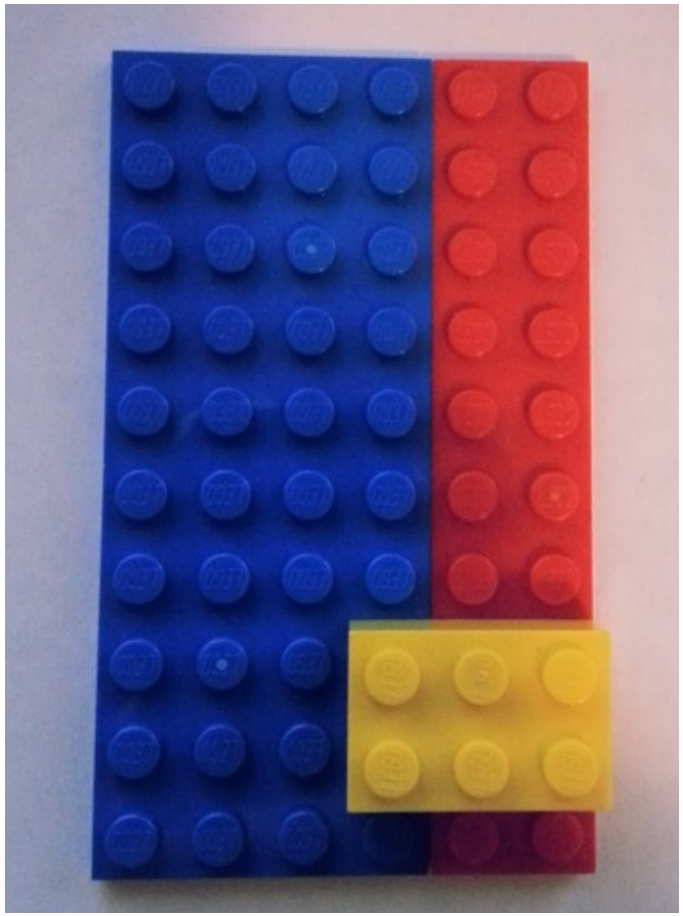
- “Common effects make particular causes less likely.”

$$\blacktriangle\ P(effect) \Rightarrow \blacktriangledown\ P(cause|effect)$$

- “More likely a cause a priori, more likely the cause a posteriori — all else equal.”

$$\blacktriangle\ P(cause) \Rightarrow \blacktriangle\ P(cause|effect)$$

The Bayesians: Lego Example



$$P(cause \mid effect) * P(effect) = P(effect \mid cause) * P(cause)$$

- **Counting in 2 ways**: a powerful tool for proofs

- 6x10 = 60 Pegs
- 4x10 = 40 **Blue** Pegs — P(**Blue**) = 40/60 = 2/3
- 2x10 = 20 **Red** Pegs — P(**Red**) = 20/60 = 1/3
- 3x2 = 6 **Yellow** Pegs — P(**Yellow**) = 6/60 =1/10

- P(**Blue**) + P(**Red**) = 1
- P(**Blue**) + P(**Red**) + P(**Yellow**) = **no go**
- P(**Yellow** | **Red**) = 4/20
- P(**Red** | **Yellow**) = 4/6
- P(**Yellow**|**Red**)*P(**Red**) = 4/20 * 20/60 = **4/60**
- P(**Red**|**Yellow**)*P(**Yellow**) = 4/6 * 6/60 = **4/60**

The Bayesians: Naïve Bayes

- We fight model combinatorial explosion with simplifying assumptions

  → More complex Bayesian models

- **Naïve Bayes:** <u>All effects are independent, given cause.</u>

  P(fever & cough | flu) = P(fever | cough, flu) * P(cough | flu)
  P(fever & cough | flu) = P(cough | fever, flu) * P(fever | flu)

  P(fever & cough | flu) **=** P(fever | flu) * P(cough | flu) **[simplify]**

  → “Having a fever doesn't change the probability of having a cough, if you have the flu.” *OR* “If you have the flu, knowing that you have a fever gives no new information.”

  - Naïve Bayes captures pairwise correlations between inputs and outputs
  - Very popular, and quite powerful: "Just a matter of counting how many times each attribute occurs with each class."

The Bayesians: Markov Models

- Next step after total independence, the bare minimum of structure

  → Vast literature on Markov models

- **Markov Property (loosely):** For a sequence, assume that the probability of the next one depends on the previous one (only)

  $P(x_n \mid x_{n-1} \leftarrow x_{n-2} \leftarrow \ldots \leftarrow x_0) = P(x_n \mid x_{n-1})$ **[notation abuse]**

- **Markov chain:** a discrete **Markov process** moving from state to state
- **Hidden Markov model (HMM):** A Markov process of observations, plus an unobservable, hidden state (process) that is dragged along

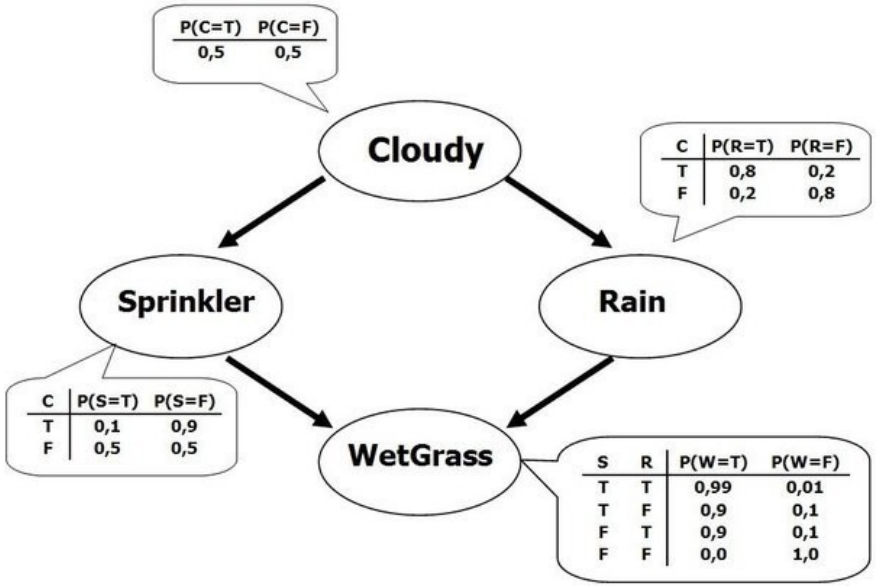
  e.g.: Model 1 : $P(\text{word}_n \mid \text{word}_{n-1})$, where word_i is a hidden state
  Model 2 : $P(\text{sound}_n \mid \text{word}_n)$

- **Kalman filter**: HMM with continuous variables, rather than discrete states.

The Bayesians: Bayesian Networks

- Judea Pearl, early 1980s: OK to have a complex network of dependencies of random variables, as long as each variable depends directly on only a few
- **Bayesian Network (BN):** Complex probability configs as **graphs** plus a probability table per variable of its parents

- Dramatic simplification, **a new language**
  - Can represent Naïve, Markovs, HMMs, etc.
- Full set of probabilities encoded into fewer values through conditional independence
- P(state) is the product of the corresponding paths through the graph
- Possible to compute P(unobserved state)
- A Bayesian network **tells a story**



The Bayesians: Bayesian Inference

$$P(hypothesis \mid data) = P(hypothesis) * P(data \mid hypothesis) / P(data)$$

- **For Bayesians, <u>learning is just another kind of inference</u>**
- **Maximum Likelihood Principle:** Of all the hypotheses available, pick the one in which seeing the data is most likely
  - Bayesians are never sure: compute posteriors for all hypotheses
  - Don’t select, entertain all hypotheses when making predictions

- Bring out the big guns
  - **Loopy belief propagation:** pretend the graph has no loops
  - **Markov chain Monte Carlo (MCMC):** do a random walk, jumping from network state to state in such a way that in the long run, each state is visited in proportion to its probability

    → MCMC can do arbitrary integral function approximation

    ⇒ Design MCMC so that its distribution converges to your target BN

The Bayesians: Case *Gen*

- **Bayesian methods are a little heavy going…**

- **Gen :** a probabilistic computing toolset built on top of Julia-lang (2019)
  - "Gen is a new probabilistic programming platform that aims to make it possible to do **real-time inference in generative models** by combining of model-based search, data-driven neural network inference, and state-of-the-art Monte Carlo techniques."
  - "Gen is thus a multi-paradigm platform for probabilistic artificial intelligence research that aims to be efficient and expressive enough for general-purpose use."
- Presentation @ Strange Loop 2019
- Paper @ PLDI 2019

The Bayesians: Conclusion

- All knowledge is uncertain, learning is a form of probabilistic inference
- **Bayes' theorem** tells us how to do inference, how to update beliefs in light of new evidence; generating the probabilities is the challenge
- **Naïve Bayes** assumes all effects are independent, given cause, capturing pairwise correlations while remaining easy to compute
- Various Markov models improve fidelity by allowing for more structure, at a computational cost; **Markov processes** are "forgetful" about path
- **Bayesian networks** capture complexities of probabilistic configurations in a convenient graphical model that tells a story
- **Markov chain Monte Carlo** (MCMC) is a powerful technique for sampling complex probability distributions and other functions
- **Bayesian learning** is about data-driven hypothesis ranking

$$P(hypothesis|data) \propto P(hypothesis) * P(data|hypothesis)$$

The Analogizers

- **Learning is about recognising similarities**
- How to determine the similarity of things?
- How to infer novel similarities?
- **The rebel tribe**
  - The least cohesive of all the tribes

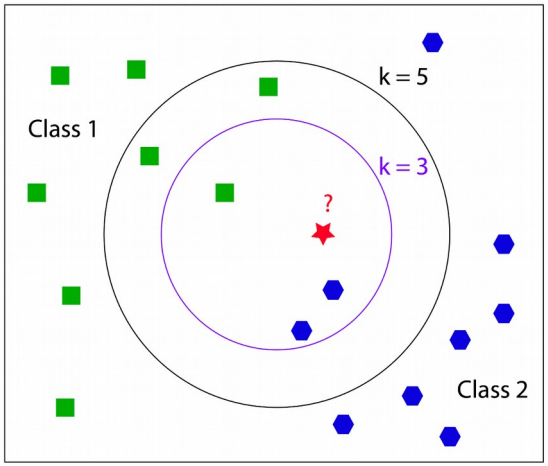
The Analogizers: Nearest-neighbour

- **Nearest-Neighbour Algorithm:** Collect data without much processing; at test time, find the item nearest to the new item
  - If nearest item meets the spec, so does this new one
  - First algorithm to be able to use an unlimited amount of data
  - Lazy and local — dynamic classification boundaries

- **K-NN:** One comparison is noisy and overfitting, so pick k items
  - Cost is detail: more voters “blurs the boundary”

- **Collaborative Filtering:** People who agreed in the past are likely to agree in the future

  → **Recommender Systems**



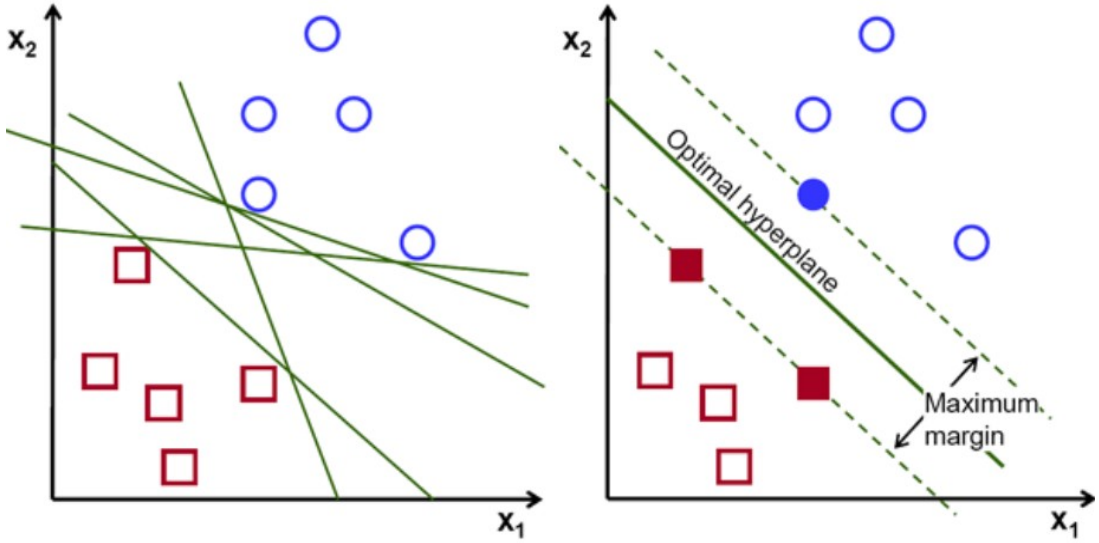
The Analogizers: In higher dimensions

- **Curse of dimensionality:** A range of issues that arise when processing data in high-dimensions spaces that do not occur in low-dimensional settings (High is thousands+, low is 2D or 3D.)
  - Trouble for nearest-neighbour!
  - All dimensions contribute to similarity measure, but most are irrelevant
  - With enough attributes, small contributions of meaningless similarities swamp out the similarity in the attributes of interest
  - More to learn in higher dimensions, more data needed for robust classification
  - Treacherous higher dimensional normal distribution

- **Blessing of non-uniformity:** Data lives in high dimensions, but is not uniformly spread out in hyperspace
  - A tiny fraction of all possible data points are reasonable, and the reasonable ones "all live together in a cozy little corner of hyperspace"

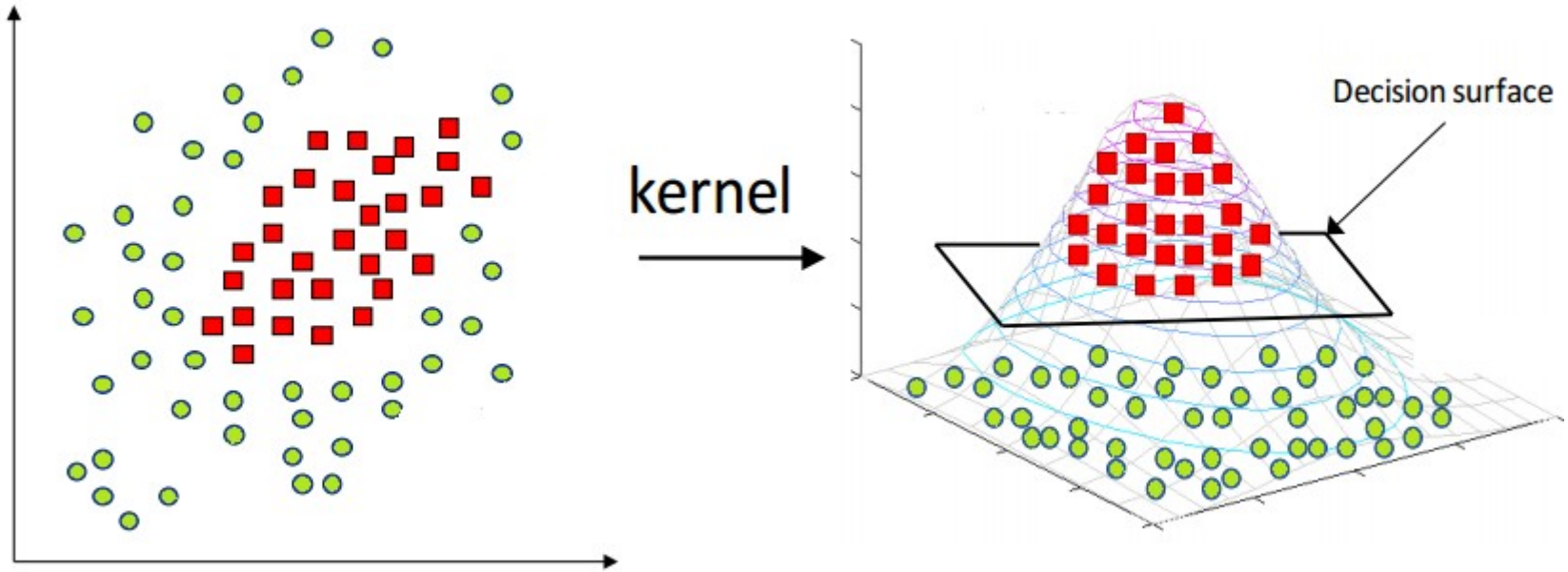
The Analogizers: Support Vector Machine

- Vladimir Vapnik, 1990s:
  How about a weighted k-NN on steroids, but not all borders are created equal?



- **Support Vector Machine**
  - Classifier frontier is determined by a set samples and weights, together with a similarity measure
  - Maximise the margin of the classification boundary
  - Can learn smooth frontiers, but needs to be constrained
  - SVM as a one hidden layer generalisation of a perceptron

The Analogizers: Kernel Trick

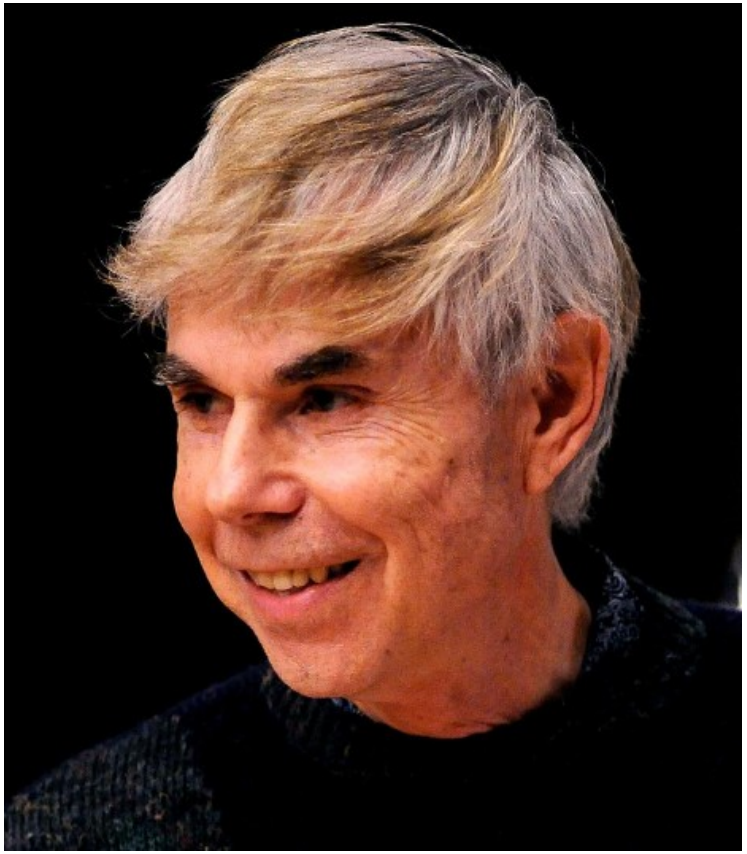


- **Kernel Trick**
  - SVMs can always create straight planes in the hyperspace, no matter how curvy the frontier may appear
  - SVMs find a max-margin hyperplane *in the kernel space* to which data is mapped from the original domain by **a kernel function**

The Analogizers: Abstract similarity

- Similarity is a spectral quality
- **Hofstadter:**
  - **Analogy**: “the fuel and fire of thinking”
  - Analogy is what the human mind does, is the fount of knowledge
- **Challenge of structural mapping**
  - Analogy is most powerful when crossing problem domain boundaries
  - Humans do it all the time, very limited success in algorithms

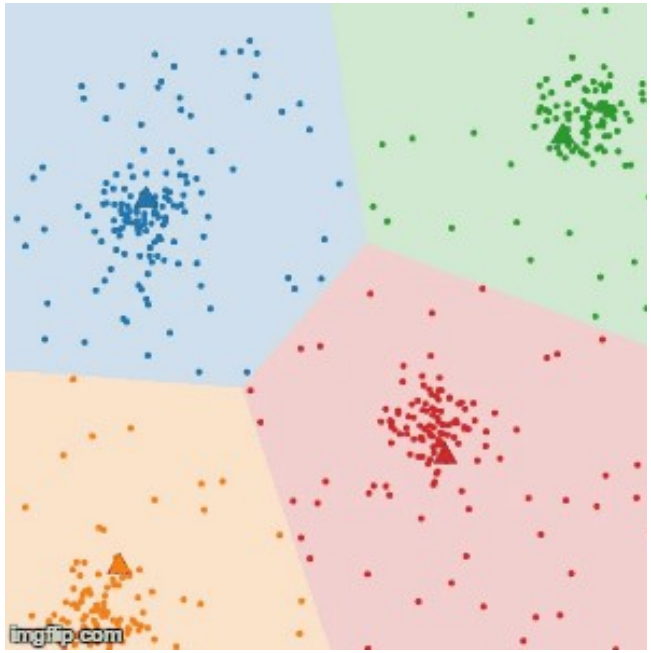
    → Knowledge Engineering
- Difficulty of faithful machine translation

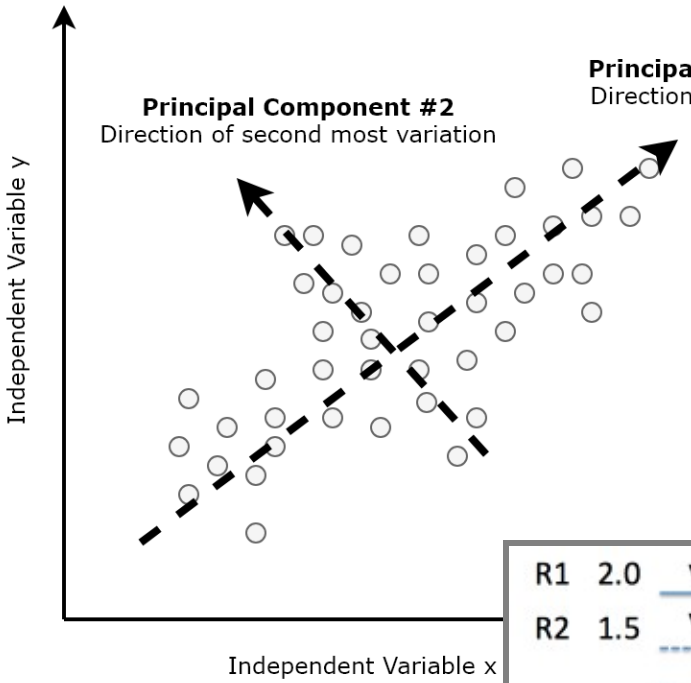


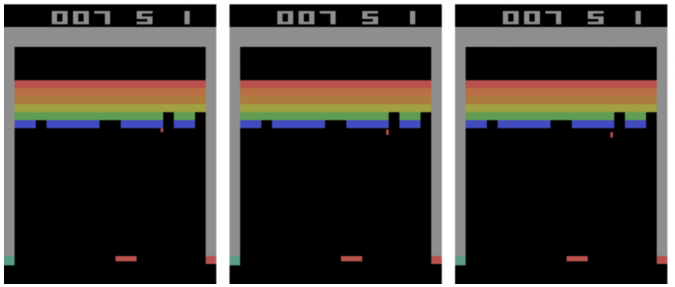
The Analogizers: Conclusion

- Similarity is a central idea in machine learning
  - Full spectrum from simple similarity to complex analogy
- In nearest-neighbour classification, a lazy local model of the data is queried with new items
  - Dynamic boundaries in arbitrarily complex concept space
- The curse of high-dimensionality, the peculiar deformations of high-dimensional hyperspace, trip up our intuition and our algorithms
  - Fortunately our data is typically non-uniform
- Support Vector Machines: maximise the margin around the boundary
- Higher abstraction similarities → analogy-making, essence of cognition

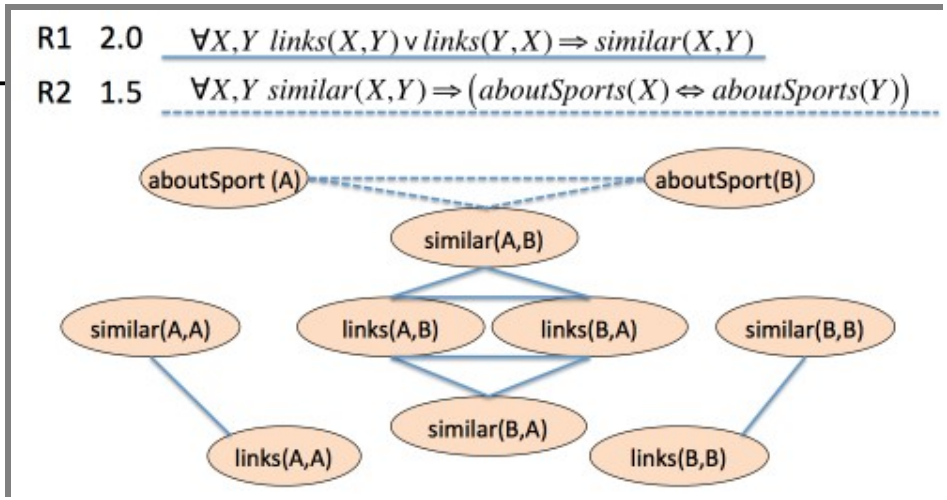
The Sixth Tribe: Self-learning







- **K-Means**
- **PCA**
- **Reinforcement Learning**
- **Relational Learning**



The Big Picture

Tribe	Origins	Master Algorithm
Symbolists	Logic, philosophy	Inverse deduction
Connectionists	Neuroscience	Backpropagation
Evolutionaries	Evolutionary biology	Genetic programming
Bayesians	Statistics	Probabilistic inference
Analogizers	Psychology	Kernel machines

Tribe	Strength	Technology
Symbolists	Structure Inference	Production Rule System Inverse Deduction
Connectionists	Estimating Parameters	Backpropagation Deep Learning
Bayesians	Weighing Evidence	HMM Graphical Model
Evolutionaries	Structure Learning	Genetic Algorithms Evolutionary Programming
Analogizers	Mapping to Novelty	kNN SVM